Candecomp/Parafac - from diverging components to a decomposition in block terms Short user guide to the Matlab codes

Alwin Stegeman †

January 14, 2016

Stegeman [2, 3] has proposed a method to deal with diverging components ("degeneracy") in real-valued three-way Candecomp/Parafac. This note contains some guidelines for the accompanying Matlab codes. The procedure is described in detail in Section 2 of [2] and Section 3 of [3]. The method is applied to a three-way dataset of TV-ratings by Stegeman [4].

We start with a CP decomposition $(\mathbf{A}, \mathbf{B}, \mathbf{C})$ featuring diverging components as described in Section 1 of [2]. This is the outcome of fitting an R-component CP decomposition to an $I \times J \times K$ data array $\mathcal{Z}$. We assume the diverging components occur because the CP problem does not have an optimal solution.

Simultaneously reorder the columns of $(\mathbf{A}, \mathbf{B}, \mathbf{C})$ such that the groups of diverging columns are next to each other. Let $d_1, \ldots, d_m$ be the numbers of columns in each group as they appear in the component matrices. Here, $d_j = 1$ corresponds to a nondiverging CP component, and we have $R = \sum_{j=1}^{m} d_j$. Let $\mathbf{L} = (d_1 \mid \ldots \mid d_m)$. The procedure works only for $\max(d_j) \leq \min(I, J, K)$ and $\max(d_j) \leq 4$. For later use, let $\mathcal{Y}$ denote the array corresponding the the CP decomposition $(\mathbf{A}, \mathbf{B}, \mathbf{C})$.

The first step of the procedure (see Section 2.1 of [2]) is to compute a block SGSD $\mathcal{Y} = (\mathbf{S}, \mathbf{T}, \mathbf{U}) \cdot \mathcal{G}$ from $(\mathbf{A}, \mathbf{B}, \mathbf{C})$. Here, the $R \times R \times R$ core array $\mathcal{G}$ is block-diagonal and block j has size $d_j \times d_j \times d_j$ and upper triangular frontal slices, $j = 1, \ldots, m$. In Matlab, computing the block SGSD is done by the command

$$[\mathtt{S}, \mathtt{T}, \mathtt{U}, \mathtt{G}, \mathtt{error\_ssq}] = \mathtt{CP2blocks}(\mathtt{A}, \mathtt{B}, \mathtt{C}, \mathtt{L});$$

where $\mathbf{G}$ denotes the matrix unfolding of the obtained block-diagonal core array $\mathcal{G}$.

Next, we compute initial values for the constrained Tucker3 model (called CP_{limit} in [4]) from

†The author is with the Heijmans Institute for Psychological Research, University of Groningen, The Netherlands, e-mail: a.w.stegeman@rug.nl, URL: http://www.alwinstegeman.nl. This research is supported by the Dutch Organisation for Scientific Research (NWO), VIDI grant 452-08-001.

the block SGSD $\mathcal{Y} = (\mathbf{S}, \mathbf{T}, \mathbf{U}) \cdot \mathcal{G}$ (see Section 2.2 of [2]). In Matlab, this is done by

$$[\texttt{A0}, \texttt{B0}, \texttt{C0}, \texttt{G0}] = \texttt{blocks2init}(\texttt{S}, \texttt{T}, \texttt{U}, \texttt{G}, \texttt{L});$$

Finally, we fit the constrained Tucker3 model (CP_{limit} in [4]) to the data array $\mathcal{Z}$ using the initial values obtained above. For this, we use the alternating least squares algorithm of Kiers & Smilde [1]. In Matlab, this is done by

$$[\texttt{SS}, \texttt{TT}, \texttt{UU}, \texttt{GG}, \texttt{X}, \texttt{Xblocks}, \texttt{iters}, \texttt{error\_ssq}] = \texttt{tucker3\_blocks}(\texttt{Z}, \texttt{A0}, \texttt{B0}, \texttt{C0}, \texttt{G0}, \texttt{conv}, \texttt{L}, \texttt{maxit}, \texttt{normalize});$$

The input parameter conv specifies the convergence criterion of the algorithm, maxit sets the maximum number of iterations, and normalize offers a choice in the normalization of the fitted constrained Tucker3 model (CP_{limit}). The output matrix $\mathbf{X}$ is the matrix unfolding of the limiting boundary point $\mathcal{X}$. The error sum-of-squares `ssq(Z-X)` should be close to but less than `ssq(Z-Y)`, indicating that $\mathcal{X}$ is indeed slightly closer to $\mathcal{Z}$ than $\mathcal{Y}$.

The files `example_I.mat` and `example_II.mat` contain the matrix unfolding $\mathbf{Z}$ of $\mathcal{Z}$, the matrix unfolding $\mathbf{Y}$ of $\mathcal{Y}$, and the CP decomposition $(\mathbf{A}, \mathbf{B}, \mathbf{C})$ for Example I and Example II, respectively, in Section 4 of [2].

References

[1] Kiers, H.A.L., & Smilde, A.K. (1998). Constrained three-mode factor analysis as a tool for parameter estimation with second-order instrumental data. *Journal of Chemometrics*, **12**, 125–147.

[2] Stegeman, A. (2012). Candecomp/Parafac - from diverging components to a decomposition in block terms. *SIAM Journal on Matrix Analysis and Applications*, **33**, 291–316.

[3] Stegeman, A. (2013). A three-way Jordan canonical form as limit of low-rank tensor approximations. *SIAM Journal on Matrix Analysis and Applications*, **34**, 624–650.

[4] Stegeman, A. (2014). Finding the limit of diverging components in three-way Candecomp/Parafac - a demonstration of its practical merits. *Computational Statistics & Data Analysis*. **75**, 203–216.